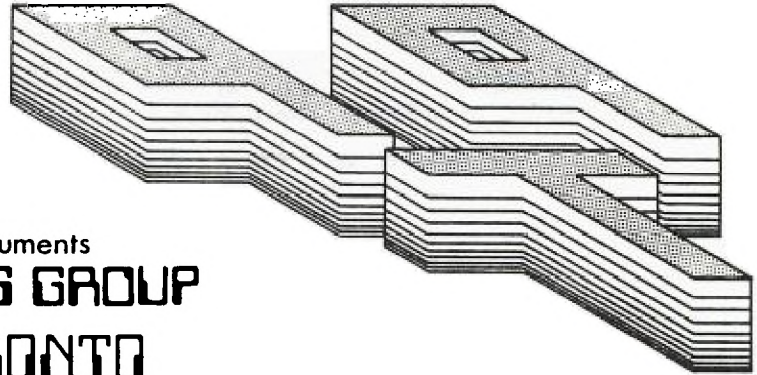




Texas Instruments  
**USERS GROUP**  
**TORONTO**

**FOR THE TI-99/4A COMPUTER**

**OCTOBER 1984**

**NINE T NINE USERS GROUP**

**55 CORDELLA AVE.  
TORONTO, ONTARIO  
M6N 2J7**

POSTED BY POST  
P.O. BOX 11100  
EDMONTON ALTA T5J 3L1



## EXECUTIVE COMMITTEE

|                  |               |            |
|------------------|---------------|------------|
| PRESIDENT        | Lloyd Lindsay | (743-3868) |
| VICE-PRESIDENT   | David Huggett | (438-4020) |
| SECRETARY        | Mike Mattos   | (763-3244) |
| TREASURER        | Gary Willert  | (276-1076) |
| OFFICER AT LARGE | Boyd Brown    | (793-3761) |

## NEWSLETTER EDITOR

David Huggett

## LIBRARY COMMITTEE

Vee. Papadimos 491-8602

## MEMBERSHIP FEES

|                      |              |
|----------------------|--------------|
| FULL MEMBERSHIP      | \$25.00/year |
| ASSOCIATE MEMBERSHIP | \$12.50/year |

All memberships are household memberships. An associate membership is only for those who live beyond the commuting distance of Toronto but who wish to receive our newsletter and have access to our library. You are welcome to visit one of our general meetings before joining the group. If you wish more information contact our secretary in writing at the club address on the front cover or call and leave a message with his answering machine.

## NEXT MEETING

Remember that the Editor's answering machine will be operative in the announce mode, after 4:00PM on the day of the meeting. This will inform members of any pertinent information should they wish to call.

The meetings are held on the last Tuesday of each month. This month's is Oct. 30 and the next one will be Nov. 27 at Shoreham Public School on 31 Shoreham Drive in Downsview, starting at 7:30PM. Shoreham Drive runs east and west off of Jane street. The entrance to the school is a few hundred yards east of Jane on the south side of Shoreham Drive.

## COMMERCIAL ADVERTISING

Any business wishing to reach our membership may advertise in our newsletter. The rates are as follows, (width by height);

|                          |         |
|--------------------------|---------|
| FULL PAGE (6" X 7.5")    | \$40.00 |
| HALF PAGE (6" X 3.5")    | \$20.00 |
| QUARTER PAGE (3" X 3.5") | \$10.00 |

Please have your ads camera ready and paid for in advance. You may contact the Editor for more information by leaving a message with his machine.

## NEWSLETTER ARTICLES

Members are encouraged to contribute to the newsletter in the form of articles, mini programs, helpful tips, jokes, cartoons or even questions. A short article may be submitted in any form but if you don't have TI-WRITER it may be better to type the longer ones into your computer using REM statements, and submit the cassette or disk, which will be returned. We welcome the reprinting of any article appearing in this newsletter providing credit is given to the author and to 9T9. If more information is required, call the Editor and please leave a message with the machine. NOTE! Long distance calls will not be returned unless a collect call is requested.

## DISCLAIMER

Opinions expressed in this newsletter are those of the writers and are not necessarily those of the 9T9 USERS' GROUP. 9T9 cannot assume liability for errors or omissions in articles, programs or advertisements.

## EDITORIAL

Welcome back! You'll notice the new format of the newsletter. I've had some complaints that the old format was difficult to read, especially the M's. This new format allows about one thousand more characters per page yet the letters should be easier to read. Remember I don't know what it will look like as I type this article. I find that the biggest limitation of this computer. More and more I long for an eighty column screen with right justification. It would be so much easier if what I saw was what I got. While I'm dreaming, it is also possible with this computer to write a word processor (in assembly of course) that would simulate typesetting. All of the letters would be proportionally spaced with the right margin justified. Well maybe someday. As you may know, I nor the rest of the executive committee are not much into playing computer games. We prefer either business oriented programs or utilities or languages that add more power to this computer. Of course most of our sample programs are universal in their applications, and should help the beginners to write better game programs. It appears though, that a large percentage of the group is into playing games. To be fair to them (and because we'd never done it before) I thought it would be a good idea to devote one club meeting in demonstrating some of the new, fast acting, bit mapped, arcade type assembly language games that are now available for our computer. This was done at the last meeting and everyone seemed to enjoy it. Well, to the other user groups that receive this newsletter, I'll bet you can empathize. I actually got two complaints from people who don't like games. Of course we can break into different interest groups which is what we usually do. (and what anyone could have done at the last meeting if they'd asked) but I've had complaints that these types of meetings are disorganized. I'll put out a general question. Is it possible to please everybody in a computer club? Speaking of complaints, we've also been criticized for not having any assembly language material in our newsletter, especially instructional or "how to" articles that make it easy to learn. Well the answer to that is easy, assembly isn't. Also, I haven't RECEIVED any such articles for publication. The popular belief seems to be that assembly can't be made simple. Well it no one else will do it, I'll give it a shot. This issue includes my first article on assembly language for beginners. Experienced programmers may note that I have perhaps oversimplified and left out a lot of information but that's the idea. If it's too difficult to start then most of us can't. The article just puts your name on the screen, then shows you how to move it. If you find that easy then try putting your name at the top and bottom of the screen and moving them into the middle. This and a counting program will be featured in next month's article. If you find that too easy, then try writing some of these articles.

DAVID HUGGETT

## CHANGING THE CASE

by DAVID HUGGETT

Have you noticed that to program in the Basic's you generally keep the alpha lock key depressed? The computer likes it that way and it will soon let you know if you try to do certain things with it up. After a while we get used to doing everything in upper case, then it becomes habit forming. Some of us will expand our systems into word processors by acquiring packages like TI-WRITER. Then this otherwise acceptable uppercase habit will begin to cause us anguish. Many of us (including myself, till I kicked the habit) have typed in whole articles on TI-WRITER and not realized until we were finished that it wasn't supposed to be in upper case. I have received several such articles from members for publication. Of all the sophisticated commands in TI-WRITER, there is nothing to change upper case into lower case, unless of course you do a "Replace/String" twenty six times. Even that isn't totally desirable. What about the capitals at the beginning of each sentence and the word "I". Well despair no longer. An unusually prolific member of our group has written a program to do that for us. It runs in Extended, but could be rewritten in Basic. It takes a TI-WRITER file from disk (line 30) and sends it where you wish, (line 20). It's in "bare bones" format and of course could be dressed up. The TI-WRITER file must have two spaces between sentences, which is the proper way to write, and the carriage return symbol must be adjacent to the period. Otherwise the next sentence will not start with a capital. There is one small bug though. It doesn't recognize proper nouns (names). I dare anyone to fix THAT bug.

CASE CHANGER  
by David Huggett

```
10 DIM T(80),B$(100)
20 OPEN #1:"PID" : or "DSK1.LOWERCASE" etc.
30 OPEN #2:"DSK2.UPPERCASE" : or whatever.
40 IF EOF(2)=0 THEN INPUT #2:A$ ELSE 60
50 S=S+1 : B$(S)=A$ : GOTO 40
60 B$(0)="."
70 FOR Y=1 TO S : M$=""
80 FOR X=1 TO LEN(B$(Y))
90 T(X)=ASC(SEG$(B$(Y),X,1)): NEXT X
100 FOR X=1 TO LEN(B$(Y))
110 IF X=1 AND
SEG$(B$(Y-1),LEN(B$(Y-1))-1,1)=". " THEN 150
120 IF X>4 THEN IF T(X-3)=46 AND T(X-2)=32 AND
T(X-1)=32 THEN 150
130 IF T(X)=73 AND T(X-1)=32 AND T(X+1)=32
THEN 150
140 IF T(X)>64 AND T(X)<91 THEN T(X)=T(X)+32
150 M$=M$&CHR$(T(X)): NEXT X
160 PRINT #1:M$ : NEXT Y
```

FOR SALE One Shugart full height, single sided, double density disk drive with power supply. Will operate outside P.E. Box. Best offer over \$200.00. York Hillary, 284-8988

## HERE'S HOW

by GARY WILLERT

Recently, I have heard a lot of words about the use of lower case letters (as opposed to the small capitals resident in the TI-99/4A). While I cannot recall ever having suffered greatly from the lack of true lower case on my machine, I am willing to concede that it might be nice to have them. It may even be that they might serve some useful function. About the only thing that comes to mind is the use of the computer by children who are not yet sure of when to use capital letters, but regardless, let's take a look at it. Lower case is not really a problem. It is merely a specific application of CALL CHAR. For the sake of saving ROM, T.I. has created the small-capital characters by applying a formula to the large-capital characters (I am not going to go into the formula just now). At power-up, a resident routine loads the characters into the appropriate locations in memory. If we want a different set of letters, then all we have to do is load them in using the CHAR subprogram. In this month's issue of HOME COMPUTER MAGAZINE, you may note an article dealing with just this subject. It is mentioned there that you can create lower-case letters by using the CHAR subprogram, but then goes on to suggest that this method is less than satisfactory because it is slow and takes up memory, and suggests the use of an assembled routine to do the job. I have no argument about the speed difference, but I would suggest that the time required is within acceptable limits; also, it is not necessary to use ANY of your program's space to redefine the characters. I wasn't particularly fond, in addition, of the idea of having to use the EDITOR/ASSEMBLER or MINI MEMORY for this purpose when most of my programs are in EXTENDED BASIC (Yes, I know we could use an assembled routine called from EXTENDED BASIC, but that would spoil the point and, besides, what about those who don't have a memory expansion?). In an earlier edition of the newsletter, I provided a listing of an auto-loading program. Based on the fact that characters are NOT redefined when we run one program from another, we can insert our character redefinition routine into this one: Voila! Any program using lower-case letters can run WITHOUT MODIFICATION and will show "true" lower-case.

By the way, "@P-" turns off the pre-scan (to save time) and "@P+" turns it back on. Everybody knows about subroutines, but there are a lot of folks who shudder at the thought of using definitions and throw up their hands in despair at trying to figure out user-written subprograms. Watch this space next issue.

```
100 OPTION BASE 1 : DIM PR$(127) : DEF X=P+(PAGE-1)*20 : CALL SCREEN(4)
110 : *****
120 : * LOWER CASE LETTERS *
130 : *****
140 :
150 CALL CHARSET : GOTO 370
160 RESTORE 190 : FOR A=97 TO 122 : READ A$
170 CALL CHAR(A,A$) : NEXT A : RETURN
180 @P-
```

```
190 DATA 0000007008384874,0040407844444478,000000
3844404438,0004043C4444443C
200 DATA 00000038447C403C,0018242070202020,000004
384438047C,0040407844444444
210 DATA 001000301010101038,0008001808084830,004040
4850704844,0030101010101038
220 DATA 0000007854545454,0000005824242424,000000
3844444438,0000007844784040
230 DATA 0000003844544834,0000005864404040,000000
3C40380478,00103810101010408
240 DATA 0000004848484824,0000004444282810,000000
4454545424,0000004428102844
250 DATA 0000004424181060,0000007C0810207C
260 :
270 : *****
280 : * AUTOLOADER *
290 : *****
300 :
310 IMAGE *****:**** SECT. AVAIL.
320 IMAGE *** *****:*** SECTORS
330 !@P+
340 DISPLAY AT(23,1):"Press ENTER to continue":"E
nter # to run"
350 ACCEPT AT(24,16)SIZE(3)VALIDATE(DIGIT):K$
360 IF K$<>" " THEN K=VAL(K$): RETURN ELSE RETURN
370 DISPLAY AT(2,1)ERASE ALL:"MENU" : D$="DSK1."
380 OPEN #1:D$:INPUT ,RELATIVE,INTERNAL : INPUT
#1:N$,A,B,C
390 DISPLAY AT(1,1):USING 310:N$,C : I=0
400 GOSUB 160
410 FOR PAGE=1 TO 7 : FOR P=1 TO 20 : I=I+1 :
IF I=127 THEN K=X : GOTO 460
420 INPUT #1:P$,A,B,C : IF LEN(P$)=0 THEN 460 EL
SE IF ABS(A)<5 THEN 420
430 DISPLAY AT(P+2,1):USING 320:X,P$,B : PR$(X)=
P$ : NEXT P : GOSUB 340
440 IF K$="" THEN CALL HCHAR(3,1,32,704)ELSE 490
450 NEXT PAGE
460 GOSUB 340
470 IF K$="" THEN CALL CLEAR : CLOSE #1 : STOP
480 IF K=0 OR K>X OR LEN(PR$(K))=0 THEN 460 ELSE
CLOSE #1
490 DISPLAY AT(14,1)ERASE ALL:"LOADING ":PR$(K)
500 CALL INIT : CALL PEEK(-31952,A,B): CALL FEE
K(A*256+B-65534,A,B)
510 C=A*256+B-65534 : A$=D$&PR$(K): CALL LOAD(C
,LEN(A$))
520 FOR I=1 TO LEN(A$): CALL LOAD(C+I,ASC(SEG$(A
$,I,1))) : NEXT I
530 CALL LOAD(C+I,0): RUN "DSKX.1234567890"
```

## TECH. TIPS

by VEE PAPADIMOS

This time I would like to cover modems, the means by which two computers talk to each other over the phone.

The word modem comes from two words, modulate and demodulate. The first part of each word gives you the word modem. The MM1 manual mini modem \$89.95 at Ace Computer on Queen street is the best deal that I have found. Most modems are in the \$300.00 starting price range. Some have auto-answer, auto originate capabilities like the Signalmann Mark III but they seem to take a little longer to "lock in" before the file transfer.

(CONTINUED AT BOTTOM OF NEXT PAGE)

# EXPERIMENTER'S CORNER

by JON BANNISTER

Have you ever wished you could restart a program after it "crashed", return to a menu after running a machine language game or peek into Command Modules and copy them? For less than \$6.00 in parts (including tax) and less than an hour's work, you can add that capability to your TI-99/4A. For those not yet familiar with the LOAD interrupt feature of the TMS9900 microprocessor chip, allow me to attempt an explanation. When the LOAD pin (pin 4 on the 9900) is brought low (set to ground) the processor performs an equivalent to a BLWP -FFFC. The processor reads memory location -FFFC, which becomes the new WP (work space pointer) and -FFFE which becomes the new PC (program counter). The program pointed to by these addresses is then executed. The only time this does not happen is if the RESET pin (pin 6) is held low (which, on the TI-99 series can only happen if you turn ON your computer or INSERT a Command Module). LOAD is a non-maskable interrupt, which means that there is no way of disabling it (theoretically). As it stands, the TI-99/4A does not support this feature. This month we shall show how you can add it. The parts you'll need for this conversion are as follows: two 6-7 inch pieces of insulated wire, .1uF bypass capacitor (Radio Shack part number 72-135 which costs \$0.99 for two), a 2.2kOhm resistor (Radio Shack part number 271-8027 or 71-1325 which costs \$0.59 for five) and a momentary contact, normally open push-button switch (Radio Shack part number 275-1547 which costs \$3.79 for five). You'll also need a small Phillips screwdriver, a soldering iron and some Rosin-core solder.

The procedure is as follows:

- 1. Read ALL of these instructions and understand them before continuing.
- 2. Take the last 1/4 inch of insulation off of each end of each piece of wire.
- 3. Tin the bared ends of the wire and the leads in the resistor with solder.
- 4. Solder the capacitor across the switch contacts using as little of the capacitor's leads as possible.
- 5. Solder one wire onto one of the switch contacts.
- 6. Solder one end of the resistor to the other switch contact.
- 7. Disconnect your Speech Synthesizer (from now on referred to as S.S.) from your computer system.
- 8. Turn the S.S. over exposing the bottom. Remove the two screws holding the case together.
- 9. Lift off the bottom of the S.S. (which is now on top). Carefully lift out the printed circuit board (PCB) with its metal enclosure.
- 10. Remove and store the two screws, washers and nuts.
- 11. Lift off the metal cover that has a hole on its top. Hold the remaining metal cover in one hand and hold the ends of the black connector in the other. Lift the connector up and away from the copper fitting.

- 12. When handling the PCB hold it only by the edges and avoid contact with any of the components as this might damage them.
- 13. Turn the PCB over to the side where no components can be seen. Orient the PCB with the black connector to your right.
- 14. Solder the wire connected to the resistor to the seventh lead from the bottom of the black connector, and the other wire to one of the 11th, 12th, 13th or 14th leads from the bottom.
- 15. Route the wires to the top of the PCB. Take the metal cover with the copper fitting and fit it over the left side of the PCB. Make sure the metal tab at the bottom of the cover is over the end of the PCB and that the wires are under the slot at the top of the cover.
- 16. Flip the board over exposing the component side. Put the remaining cover on the board. Replace the screws, washers and nuts (the washers go under the nuts, not the screw). Tighten the screws. (Remember you're not changing a tire - take it easy.)
- 17. (Before you drill any holes, place the PCB back into the black box and check for clearance between the switch and PCB.) Measure 3/8 of an inch over from the side of the S.S.'s black box at the end with the silver top. Measure down from 3/8 to 1 inch. At the cross of these two markers drill a 5/16 inch hole.
- 18. Fit the switch through the hole. Install the washer and nut and tighten.
- 19. Place PCB into black box (it only goes one way). Put the bottom back on, replace and tighten the screws. Connect the S.S. back to your computer and you're done!

Some hints on how to use this switch: When using the Editor/Assembler Debug program the Debugger automatically sets -FFFC and -FFFE. If it ever crashes just push the button. From Extended Basic, Minmemory or E/A Basic you can do a reset with a CALL LOAD(-4,131,224,0,36). To peek into cartridges, load Debug and run it. Remove the cartridge and replace it with the one you wish to play with and push the button - voila! In my next article I'll show how you can explore into cartridges by removing the RESET on the cartridge slot. Disclaimer: This modification will void the manufacturer's warranty. Neither I, nor the 9T9 Toronto Users Group can be held responsible for errors, omissions or any damage occurring during or after this modification.

## (TECH. TIPS CONTINUED)

Smart modems are capable of auto-dial and auto-answer but as the features increase, so does the price. I own the MM1 and it's all you'll probably ever need. Here are its pin connections.  
Pin 2 of the RS232 to pin 3 of the modem  
Pin 3 of the RS232 to pin 2 of the modem  
Pin 5 of the RS232 to pin 5 of the modem  
Pin 20 of the RS232 to pin 6 of the modem  
Pin 7 of the RS232 to pin 7 of the modem  
Pin 8 of the TI RS232 to pin 8 of modem  
In future articles I will be covering the microfazer and how to repair your memory card at home.

## EASY TO ASSEMBLE

by DAVID HUGGETT

Assembly is the ultimate language of any computer as far as accessing the full speed and power is concerned. Even if you don't wish to become a dedicated assembly programmer it is desirable to know something about it, even if it is only to write simple super fast routines to be assessed from Extended Basic or even Forth. You are in complete control of the computer with assembly. Not only will your programs run many hundreds of times faster (it executes about one to two hundred thousand instructions per second, depending on which one you use) but you can do anything, many of which are impossible in the Basic's. Like keeping any part or parts of the screen still while scrolling the other parts, or working out and displaying numbers to any amount of decimal places, using a forty column window on any size screen, incredible animated graphics, etc. Because of this power, assembly is more detailed and you have to know more about the computer to use it, which is why few people attempt to learn it. Also the books available that teach it are usually written by computer experts some of whom probably speak assembly better than English. Every book that I have read so far starts out with charts and schematics and goes into depth on hexadecimal to binary to decimal number conversions, then esoteric chapters on CPU registers, instruction sets, addressing formats, editor fields ..... Forget it! Lets just start programming and if you're an above average mumbly like me, "all that other stuff'll come naturley"

Plug in your Editor/Assembler module, insert the E/A diskette part A, press 2 for Editor Assembler, then 1 for the edit mode and 2 to edit. It only takes a few seconds and we're ready to write our first assembly program. Make sure you have the feel of the editor. Use your E/A overlay strip and just bang around a bit. Type in some words, press enter, try the insert and delete keys, use the tabs etc. You'll notice you can show line numbers by going to the extreme left margin with the left arrow key. If you're really excited, after you have several lines of text you can push FCTN 9 (escape) and play with the MOVE, COPY, REPLACE and other arcane features, but we don't need that stuff yet. So delete what's on your screen (FCTN 3) and let's get started.

We'll start with the following program. All that it does is put a name on the screen. You'll notice that the first two lines are indented from the third one. Just use the tab (FCTN 7) and your program will line up like mine. The first line just DEFINES the name of the program and the second line makes a REFERENCE to the built in utility that we are going to use, the Video Multiple Byte Write. The program really starts in line three which I have labelled START. I could call it anything I liked as long as the same name was in the DEFINITION table (that's line one). LI means Load Immediate but I call it Load Into. So line three says, Load Into Register Zero the value 365. Register Zero is where the printing starts on the screen. Instead of Rows and Columns, Assembly just uses 768

screen locations so 365 will be somewhere in the middle of the screen. It would help if you printed out a chart with all of the screen locations, 0 to 31, 32 to 63, etc. Register One is always for WHAT is to be printed. In this case we use the label NAME. We could call it whatever we wished as long as the TEXT line started with the same label. You can change the TEXT line between the single quotes to your own name, but you must remember to count the letters in your name, including spaces, and put that number into register two. This register always holds the number of bytes we are writing to the screen. The sixth line just prints the information above it, (in the three registers) to the screen. The last line, END isn't even necessary. It just tells the assembler to stop assembling. It would stop anyway if it had nothing to assemble but we'll use it to practice good programming habits. You also don't need the lower case comments but they're recommended by the experts. But suit yourself, they can be typed any way you wish, upper case etc, as long as there is at least one space between them and the actual program. That's it, except that the computer won't understand any of this. What we have to do is to convert this SOURCE program into numbers because that's all that the computer understands, unless it's written in Basic, but then the computer still converts it into numbers after every line as it's running, and we know how slow that is. If we convert it first then the computer can devote all of its time to running it. This conversion process is called assembling the program into a pure number program, called the object program, or object code. Then we run the object code. We keep the source code just in case we wish to make any changes. Now we must save our source code. Get out of the editor by pressing FCTN 9 (you may have to press it twice depending where you are now) and then press 3 for SAVE. The source code is always a display variable 80 file and the object code is display fixed 80. Since this is source code press Y for yes then type your file name, say DSK1.NAME. After it's saved press FCTN 9 to escape back to the master selection list and chose 2 for assemble then press Y to load the assembler. ENTER the name of the source code, eg. DSK1.NAME then ENTER the name that you want your object code called, say DSK1.NAMEOB. For LIST file just press enter and press R for options, then enter and your program is assembled. It only takes a few seconds, then press enter and FCTN 9. Now select 3 for Load and Run and ENTER the name of the object file. After it loads press enter again then type the program name, which we called START. It will run instantly. That's all there is to it. It really is easier than it sounds.

```
DEF  START      this is the program name.
REF  VMBW       put VMBW in reference table.
START LI  R0,365 screen location (or 16D).
      LI  R1,NAME what is to be written.
      LI  R2,7   how many bytes to write.
      BLWP @VMBW write above to the screen.
NAME  TEXT 'MY NAME'
      END
```

Notice in the line labelled START that you can use either 365 or HEX 16D. The less than sign means Hexadecimal. They both equal the same

thing, just in different number systems. Our assembler doesn't care what you use, if it is decimal it will automatically convert it to hexadecimal and the object codes will be identical. There is a small catch though. When using VSBW, that is if you are writing a single byte, it only reads the left two digits of a four digit hexadecimal number, which is the most significant byte (MSB). Therefore the space character (ASCII 32) is >20, which in this case, should be written as >2000. To use decimal, it would have to be converted to 8,192 which is more awkward than hexadecimal. I would suggest that you get a decimal to hex conversion chart, or a calculator that converts them. It gets easy with practice, though.

Now that you have run your program you might wonder what good is all that speed and power if we just put our name on the screen. Well maybe if we moved it from one screen position to another we could see the speed. The only problem is that assembly is so fast that it can put your name into all 768 screen locations faster than your eyes could see what was happening. So we have to write a delay loop. Look at the program at the end of this article, the line labelled DELAY. After each time your name is displayed the computer counts to 700. To do this it executes 2,100 instructions. The program is almost self explanatory. We have used a couple of new instructions. INC just means to increment the register by one. CI means to compare the register with whatever value follows it. JLT (jump if less than), JEQ (jump if equal), JMP (jump no matter what) etc. They are all listed in the manual. The three lines that start with the label SCAN work like a CALL KEY in basic. Forget about how it works, just use those three lines when you want the computer to wait for any key press before going on to the next line. The last question you might have is why do we write a space character to the screen every time before your name? Take it out and see what happens. That's another advantage of assembly, you don't have to clear the whole screen every time, just clear the spots with stuff in them. That's what that does. As your name moves across the screen it overwrites the old name, except for the first character, which we cleared with the space character. Not so tough after all, is it?

```

DEF  START      name of program
REF  VMBW,VSBW,KSCAN  utilities used
START  LI  R0,0  beginning screen location
LOOP  LI  R1,12000  the space character in Hex
      BLWP @VSBW  write it to the screen
      INC R0  add one to screen location
      LI  R1,NAME  what is to be printed
      LI  R2,7  how many bytes to print
      BLWP @VMBW  write them to the screen
      LI  R3,0  start register 3 at zero
DELAY  INC R3  add one to R3 (increment)
      CI  R3,700  compare R3 to 700
      JLT DELAY  if less, jump to DELAY
      CI  R0,760  is NAME near end of screen
      JLT LOOP  if not, jump to LOOP
CAN  BLWP @KSCAN  if yes, scan keyboard
      MOVE @B37C,R0  look for a keypress
      JEQ SCAN  if none, jump to SCAN
      JMP START  if yes, then start again
NAME  TEXT MY NAME  what is printed to screen
      END

```

## BACK TO BASICS

by DAVID HUBGETT

Have you ever become discouraged with your computer? After the initial thrill of having a computer in the house, learning some of the basic basics, and starting to feel quasi computer literate, you come to a few hurdles that appear impassable. So the computer is used more for games than learning how to program, then it's used less and less. Also it's such a nuisance having to load in long programs to really do anything. Well that doesn't always have to be the case. Every once in a while I have had a question that would have taken up to an hour or more to solve with a calculator. I was able to type in a short program in a few minutes that gave me the answer in seconds. The following is such an example. With the trust companies vying for our business they appear to be offering more lucrative policies. One of these was a daily interest rate. I wanted to see how much more that was over a yearly interest rate. It is simple to figure out the yearly rate. \$1000.00 at 10% makes \$100. But daily interest is more difficult. You are making interest on your interest, so you must figure out your interest every day. That's done by taking 10% of your \$1000.00, dividing that by 365, then adding it to the \$1000.00. Then you take the new balance and repeat the process 365 times. The program below does that in less than 10 seconds (extended basic). Line 10 is your balance. Line 30 does the calculations. Notice that I have used 10%, that's the .1. Use whatever you wish, 11% = .11 etc. Lines 20 and 40 are the loop so that the calculations can be done for every day of the year. Line 50 rounds off the remainder to the nearest cent and prints the new balance. The program is in "bare bones" format, you may wish to dress it up, but other than figuring out your daily interest it just tells us one thing. That 10% daily interest equals 10.516% yearly interest. Humm.... Interesting.

```

10 B=1000
20 FOR X=1 TO 365
30 B=(B*.1/365)+B
40 NEXT X
50 PRINT INT((B+.005)*100)/100

```

## LIBRARY PROGRAMS

We are finally getting our library organized. With new programs coming in all the time it has become a perpetual task keeping it organized, with the credit going to the library members. They have finally put together a library listing which includes the necessary information to help you select your programs. Should you wish to contact them, they are listed below.

Bruce McDonald, 3753 Teeswater Rd. 577-5385

Nicos Eydemon, 72 Southwood Dr. 360-6844

Wes Shunk, 2 Quarry Crt. 464-1018

## ADMENDENTS

The following amendments to the Operating Procedures shall be tabled at the forthcoming Annual Meeting, scheduled for 29 JAN 85. The following changes are proposed in order to add flexibility to the executive so that specific officers are not tied to certain committees. For instance, it may be more advantageous to the group for the officer at large to be on the library committee rather than the equipment committee.

### ARTICLE 1, PARAGRAPH 1

From:

The president shall preside at all general meetings and all executive officers meetings of the User's Group. shall be a member of the Program Committee, and shall perform all such duties as are incidental to the office and are properly required of the chief executive officer.

To:

The president shall preside at all general meetings and all executive officers meetings of the User's Group and shall perform all such duties as are incidental to the office and are properly required of the chief executive officer.

### ARTICLE 1, PARAGRAPH 2

From:

The vice president shall, in the absence of the president, exercise all the functions of, and shall be vested with all the powers of the president, and shall be a member of the education committee, and shall fulfill the function of editor of the newsletter.

To:

The vice president shall, in the absence of the president, exercise all the functions of, and shall be vested with all the powers of the president.

### ARTICLE 1, PARAGRAPH 4

From:

The secretary shall have charge of all papers, keep such records, make such reports, and perform all such duties as are incidental to the office and are properly required of the secretary, and shall be a member of the Library Committee, shall be the officer responsible for co-ordinating and monitoring the activities and procedures of the various special-interest groups that may exist from time to time.

To:

The secretary shall have charge of all papers, keep such records, make such reports, and perform all such duties as are incidental to the office and are properly required of the secretary.

### ARTICLE 1, PARAGRAPH 6

From:

The officer at large shall perform those duties that may be assigned to him by the president, and shall be a member of the Equipment Committee.

To:

The officer at large shall perform those duties that may be assigned to him by the president.

Last year, we experienced a great deal of difficulty in budgeting during the first part of the year because a large number of members did not renew until April, as was their privilege under the current Operating Procedures. In order to encourage people to renew earlier, the executive have announced an Early-Bird Sale and propose the following ammendment.

### ARTICLE 4, PARAGRAPH 3, SECTION b

From:

Members shall be suspended when dues are three (3) months or more in arrears and may, at the discretion of the Executive Committee, be dropped from the membership.

To:

Members shall be suspended when dues are one (1) month or more in arrears and may, at the discretion of the Executive Committee, be dropped from the membership.

For your convenience, we are reprinting here ARTICLE 7, PARAGRAPH 1, which deals with amendments to the Operating Procedures:

"These Operating Procedures may be ammended by a two-thirds (2/3) vote of th active members in good standing who are present in person or represented by proxy at any regular meeting, but no ammendment shall be voted upon without two (2) weeks written notice to the members in good standing."

Remember that you cannot vote unless you are either present in person or represented by proxy. Proxy forms will be available at the November and December meetings, but a signed note will do.

## EARLY BIRD SALE !!!

The 1985 membership fees are as follows:

|                      |         |                               |
|----------------------|---------|-------------------------------|
| FULL MEMBERSHIP      | \$25.00 | (\$20.00 for charter members) |
| ASSOCIATE MEMBERSHIP | \$12.50 | (\$10.00 for charter members) |

For those who have questions about the different types of membership:

Full membership carries all Group privileges. Associate membership is for our out-of-town members. There have been a number of questions regarding associate memberships during the past year. You qualify for this only if a telephone call to metro Toronto is long distance. Associate members have all privileges except the right to vote at meetings.

Charter members are those who joined during the first membership drive, when the Group had very little to offer as incentive, and who have continuously been members ever since. Charter members who let their dues fall into arrears will lose their charter status.

A \$5.00 reduction on full memberships and \$2.50 on associate memberships will apply to all memberships, new or renewal, paid between now and 29 JAN 1985.

Your Group membership is a real bargain anytime and now it is an even better buy - but only till JAN 29.

GARY WILLERT

TREASURER/MEMBERSHIP CHAIRMAN